WHAT IS CLAIMED IS:

1. A software package verification tool for verifying a software package that includes at least one software component, the tool comprising:

a framework operable to identify at least one test module defining a test of at least one parameter of the at least one software component of the package; and

a control module operable to access the framework to cause the at least one test module identified therein to perform the test defined thereby for verifying the package.

2. The tool of claim 1, wherein the framework identifies a plurality of test modules.

3. The tool of claim 2, wherein the framework identifies a priority for each of the test modules.

4. The tool of claim 3, wherein the control module is operable to cause the test modules to be executed sequentially according to the priority identified in the framework for each of the test modules.

5. The tool of claim 1, wherein a mechanism is provided for identifying the at least one test module as being one of active and not active.

6. The tool of claim 5, wherein the mechanism for identifying the at least one test modules as being one of active and not active is included in the framework.

7. The tool of claim 5, wherein the mechanism for identifying the at least one test modules as being one of active and not active is included in the control module.

8. The tool of claim 2, wherein the framework comprises a directory having a

plurality of entries, each entry identifying one of the plurality of test modules.

9. The tool of claim 8, wherein each entry defines a priority for the one of the test modules identified therein.

10. The tool of claim 8, wherein the identity of the one of the test modules defines its priority.

11. The tool of claim 2, wherein each of the plurality of test modules is formed by a script and the framework identifies each of the test modules by a name for the script.

12. The tool of claim 2, wherein each of the test modules is formed by a software object.

13. A computer program on a carrier medium for verifying a software package that includes at least one software component, the computer program comprising computer executable instructions:

a) forming a framework operable to identify at least one test module defining a test of at least one parameter of the at least one software component of the package; and

b) forming a control module operable to access the framework to cause the at least one test module identified therein to perform the test defined thereby for verifying the package.

14. A program storage device readable by a computer, tangibly embodying a program of instructions executable by the computer to perform method steps for verifying a software package that includes at least one software component, the method comprising the steps of:

a) providing a framework for identifying at least one test module, each said test module defining a test of at least one parameter of the at least one software component of the package;

b) accessing the framework to identify the at least one test module; and

c) causing the at least one test module to perform the test defined thereby on the package.

15. The method of claim 14, wherein the framework identifies a plurality of the test modules.

16. The method of claim 15, wherein a priority for each of the test modules is identified in the framework.

17. The method of claim 15, comprising sequentially causing each of the test modules to be executed according to the priority identified for each of the test modules.

18. The method of claim 15, comprising identifying each of the test modules as being one of active and not active.

19. The method of claim 15, comprising providing a directory in the framework, wherein the directory has a plurality of entries, each entry identifying one of the plurality of test modules.

20. A system for verifying a software package that includes at least one software components, the system comprising:

a) a framework to identify at least one test module defining a test of at least one parameter of the at least one software components of the package; and

b) a control module operable to access the framework for causing the at least one test module identified therein to perform the test defined thereby for verifying the package.

21. The system of claim 20, wherein the system comprises a computer including a processor, memory and software held in the memory and operable to

control the processor, the software forming:
said framework and said control module.

22. A computer system for verifying a software package that includes at least one software component, the system comprising:

a) a memory for storing software; and

b) a processing unit for executing the software to carry out the steps of:

(i) providing a framework to identify at least one test module defining a test of at least one parameter of the at least one software component of the package; and

(ii) providing a control module operable to access the framework for causing the at least one test module identified therein to perform the test defined thereby for verifying the package.

23. A method of verifying a software package that includes at least one software component, the method comprising:

a) providing a framework for identifying at least one test module, each said test module defining a test of at least one parameter of the at least one software component of the package;

b) accessing the framework to identify the at least one test module; and

c) causing the at least one test module to perform the test defined thereby on the package.

24. The method of claim 23, wherein the framework identifies a plurality of test modules.

25 The method of claim 24, wherein a priority for each of the test modules is identified in the framework.

26. The method of claim 25, comprising sequentially causing the test modules to be executed according to the priority identified for each of the test modules.

27. The method of claim 24, comprising identifying each of the test modules as being one of active and not active.

28. The method of claim 24, comprising providing a directory in the framework, wherein the directory has a plurality of entries, each entry identifying one of the test modules.

29. The method of claim 28, wherein each entry defines a priority of the test module identified thereby.

30. The method of claim 28, wherein the identity of a module defines its priority.

31. A method of verifying a software package that includes at least one software component, the method comprising:

a) receiving the software package;

b) accessing a framework that references at least one test module, each said test module defining a test of at least one parameter of the at least one software component of the package, for identifying the at least one test module from the framework; and

c) performing the test defined by the at least one test module on the package.

32. The method of claim 31, including repeating steps (b) and (c) to perform a sequence of tests, the order in which the tests are performed being determined by relative priorities assigned to each of the at least test module.

33. A computer readable medium having stored thereon a data structure operable for use in verifying a software package that includes at least one software component, the data structure comprising:

a) a first data field containing data representing one of a plurality of test modules, each test module being operable to test at least one parameter of the at least one software component of the package,

b) where data representing ones of the test modules may be added to and deleted from the data structure, creating a flexible test structure.

34. The medium according to claim 33, wherein the data structure further comprises a second data field identifying a priority for each of the test modules represented by the data in the first data field, the priority defining an order of execution of the test modules.

35. The medium according to the claim 33, wherein the data structure further comprises a third data field identifying the one of a plurality of test modules represented by the data in the first data field as being one of active and not active.